

日 本 国 特 許 庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日 2 0 0 3 年 1 1 月 2 5 日
Date of Application:

出 願 番 号 特 願 2 0 0 3 - 3 9 3 4 1 6
Application Number:

[ST. 10/C]: [J P 2 0 0 3 - 3 9 3 4 1 6]

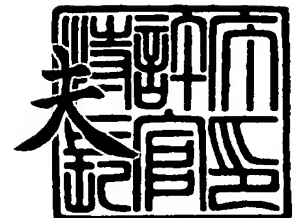
出 願 人 株式会社リコー
Applicant(s):



2 0 0 3 年 1 2 月 1 8 日

特許庁長官
Commissioner,
Japan Patent Office

今 井 康



【書類名】 特許願
【整理番号】 0308738
【提出日】 平成15年11月25日
【あて先】 特許庁長官 今井 康夫 殿
【国際特許分類】 G03G 21/00
G06F 17/60
【発明者】
【住所又は居所】 東京都大田区中馬込 1 丁目 3 番 6 号 株式会社リコー内
【氏名】 小林 綾子
【特許出願人】
【識別番号】 000006747
【氏名又は名称】 株式会社リコー
【代理人】
【識別番号】 100070150
【弁理士】
【氏名又は名称】 伊東 忠彦
【先の出願に基づく優先権主張】
【出願番号】 特願2002-342826
【出願日】 平成14年11月26日
【手数料の表示】
【予納台帳番号】 002989
【納付金額】 21,000円
【提出物件の目録】
【物件名】 特許請求の範囲 1
【物件名】 明細書 1
【物件名】 図面 1
【物件名】 要約書 1
【包括委任状番号】 9911477

【書類名】 特許請求の範囲**【請求項 1】**

画像形成処理で使用されるハードウェア資源と、画像形成に係る処理を行うプログラムとを有する画像形成装置であって、

前記ハードウェア資源に関するチェックを行うチェック処理と前記プログラムとの関連を設定する設定手段と、

前記チェック処理を行い、そのチェック処理の結果に応じて前記チェック処理に関連する前記プログラムを起動する起動手段と

を有することを特徴とする画像形成装置。

【請求項 2】

前記設定手段は、前記チェック処理と前記プログラムとを 1 対 1 に関連付けて設定していることを特徴とする請求項 1 記載の画像形成装置。

【請求項 3】

前記設定手段は、前記チェック処理と前記プログラムとを 1 対 n に関連付けて設定していることを特徴とする請求項 1 記載の画像形成装置。

【請求項 4】

前記設定手段は、前記チェック処理と前記プログラムとを n 対 1 に関連付けて設定していることを特徴とする請求項 1 記載の画像形成装置。

【請求項 5】

前記チェック処理の結果を格納しておく格納手段を更に有し、

前記起動手段は、前記ハードウェア資源に関するチェックを行う前に前記格納手段に格納されているチェックの結果を確認し、これから行うチェックの結果が格納済みであれば前記チェックの結果を利用することを特徴とする請求項 1 記載の画像形成装置。

【請求項 6】

前記起動手段は、前記設定手段に設定されている内容に応じて前記チェック処理を行うことを特徴とする請求項 2 乃至 5 何れか一記載の画像形成装置。

【請求項 7】

前記起動手段は、前記プログラムを起動した後に、前記チェック処理を終了させることを特徴とする請求項 6 記載の画像形成装置。

【請求項 8】

前記起動手段は、チェック処理を行って前記ハードウェア資源の有無をチェックし、前記ハードウェア資源が有ると判定したときに正常値をチェックの結果として取得し、前記ハードウェア資源が無いと判定したときに異常値をチェックの結果として取得することを特徴とする請求項 2 乃至 7 何れか一項記載の画像形成装置。

【請求項 9】

前記起動手段は、前記チェック処理として前記ハードウェア資源に対応したデバイスドライバをオープンする為の処理を行い、オープンが成功したとき又は既にオープンされていたときに前記ハードウェア資源が有ると判定し、それ以外のときに前記ハードウェア資源がないと判定することを特徴とする請求項 8 記載の画像形成装置。

【請求項 10】

前記起動手段は、プリンタ、コピー、ファックス又はスキャナの一部又は全部として動作するハードウェア資源の有無のチェックの結果として正常値を取得したときに、前記設定手段に設定されている内容に応じてプリンタ、コピー、ファックス又はスキャナの処理を行う前記プログラムを起動することを特徴とする請求項 8 又は 9 記載の画像形成装置。

【請求項 11】

前記起動手段は、チェック処理を行って前記ハードウェア資源の有無をチェックし、前記ハードウェア資源が無いと判定したときに正常値をチェックの結果として取得し、前記ハードウェア資源が有ると判定したときに異常値をチェックの結果として取得することを特徴とする請求項 2 乃至 7 何れか一項記載の画像形成装置。

【請求項 12】

前記起動手段は、ハードディスク装置の有無のチェックの結果として正常値を取得したときに、前記設定手段に設定されている内容に応じてRAMディスクをマウントすることの特徴とする請求項11記載の画像処理装置。

【請求項13】

前記起動手段は、前記チェック処理として前記ハードウェア資源の所定の性能をチェックし、前記ハードウェア資源が所定の性能を満たしていると判定したときに正常値をチェックの結果として取得し、前記ハードウェア資源が所定の性能を満たしていないと判定したときに異常値をチェックの結果として取得することの特徴とする請求項2乃至7何れか一項記載の画像形成装置。

【請求項14】

前記起動手段は、中央演算処理装置(CPU)の性能のチェックの結果として正常値を取得したときに、前記設定手段に前記チェック処理と関連付けて設定されている前記プログラムを起動し、前記チェックの結果として異常値を取得したときに、前記設定手段に前記チェック処理と関連付けて設定されている前記プログラムを起動しないことの特徴とする請求項13記載の画像形成装置。

【請求項15】

前記起動手段は、メモリの性能のチェックの結果として正常値を取得したときに、前記設定手段に前記チェック処理と関連付けて設定されている前記プログラムを起動し、前記チェックの結果として異常値を取得したときに、前記設定手段に前記チェック処理と関連付けて設定されている前記プログラムを起動しないことの特徴とする請求項13記載の画像形成装置。

【請求項16】

前記設定手段は、前記チェック処理と前記プログラムの存在するディレクトリ又は上位のディレクトリとを関連付けて設定しており、

前記起動手段は、前記チェックの結果が正常値であるときに前記チェック処理に関連付けて設定されている前記ディレクトリをマウントし、前記チェックの結果が異常値であるときに前記チェック処理に関連付けて設定されている前記ディレクトリをマウントしないことの特徴とする請求項14又は15記載の画像形成装置。

【請求項17】

前記起動手段は、チェック処理を行って前記ハードウェア資源に格納されている所定の識別子をチェックし、前記所定の識別子が所定の条件を満たしていると判定したときに正常値をチェックの結果として取得し、前記所定の識別子が所定の条件を満たしていないと判定したときに異常値をチェックの結果として取得することの特徴とする請求項2乃至7何れか一項記載の画像形成装置。

【請求項18】

前記起動手段は、チェック処理を行ってSDカードに格納されている識別子をチェックし、その識別子が前記SDカードが挿入されているスロットの識別子と一致していると判定したときに正常値をチェックの結果として取得し、前記識別子が前記SDカードが挿入されているスロットの識別子と一致していないと判定したときに異常値をチェックの結果として取得することの特徴とする請求項17記載の画像形成装置。

【請求項19】

前記起動手段は、前記チェックの結果として正常値を取得したときに、前記設定手段に前記チェック処理と関連付けて設定されている前記プログラムを実行し、前記チェックの結果として異常値を取得したときに、前記設定手段に前記チェック処理と関連付けて設定されている前記プログラムを実行しないことの特徴とする請求項17又は18記載の画像形成装置。

【請求項20】

前記格納手段は、前記起動手段が直接アクセス可能なメモリ上の領域に作成されることの特徴とする請求項5記載の画像形成装置。

【請求項21】

前記起動手段は、電源投入後に起動されたオペレーティングシステムにより起動されることを特徴とする請求項1記載の画像形成装置。

【請求項22】

前記プログラムは、画像形成処理を行うアプリケーションのプログラムと、画像形成処理で利用されるハードウェア資源の管理を行うコントロールサービスのプログラムと、オペレーティングシステムのプログラムとを有することを特徴とする請求項1記載の画像形成装置。

【請求項23】

画像形成処理で使用するハードウェア資源と、画像形成に係る処理を行うプログラムとを有する画像形成装置のプログラム起動方法であって、

起動手段が、前記プログラムと前記ハードウェア資源に関するチェックを行うチェック処理との関連を設定した設定手段を解析する解析段階と、

前記起動手段が、前記解析の結果に応じて前記チェック処理を実行するチェック処理実行段階と、

前記起動手段が、前記チェック処理の結果に応じて前記チェック処理に関連する前記プログラムを起動するプログラム起動段階と

を有することを特徴とするプログラム起動方法。

【請求項24】

画像形成処理で使用するハードウェア資源と、画像形成に係る処理を行うプログラムとを有するコンピュータを、

前記ハードウェア資源に関するチェックを行うチェック処理と前記プログラムとの関連を設定する設定手段と、

前記チェック処理を行い、そのチェック処理の結果に応じて前記チェック処理に関連する前記プログラムを起動する起動手段と

して機能させるためのプログラム起動プログラム。

【書類名】明細書

【発明の名称】画像形成装置、プログラム起動方法及びプログラム起動プログラム

【技術分野】

【0001】

本発明は、画像形成装置、プログラム起動方法及びプログラム起動プログラムに係り、特に所定の設定ファイルに応じてプログラムを起動するプログラム起動方法、そのプログラム起動方法を利用する画像形成装置及びプログラム起動プログラムに関する。

【背景技術】

【0002】

近年、プリンタ、コピー、ファクシミリおよびスキャナなどの各装置の機能を1つの筐体内に収納した画像形成装置（以下、融合機という）が知られるようになった。この融合機は、1つの筐体内に表示部、印刷部および撮像部などを設けると共に、プリンタ、コピー、ファクシミリおよびスキャナにそれぞれ対応する4種類のソフトウェアを設け、そのソフトウェアを切り替えることより、プリンタ、コピー、ファクシミリおよびスキャナとして動作させるものである。例えば特許文献1には、融合機の一例が記載されている。

【0003】

このような融合機は、電源投入後に、BIOS (Basic Input/Output System) およびブートローダ (Boot Loader) が起動する。ブートローダは、カーネル (Kernel) およびルートファイルシステムをRAM (Random Access Memory) 上に展開してカーネルを起動する。そして、カーネルはルートファイルシステムをマウントする。ここでマウントとは、ファイルシステムや周辺機器などをアクセス可能な状態に起動することをいう。

【0004】

カーネルの起動後、アプリケーション（以下、アプリという）や各種サービスを起動するアプリ／サービス層起動プログラムが起動される。アプリ／サービス層起動プログラムは融合機で最初に起動されるプロセスであり、所定の設定ファイルに従ってファイルシステムをマウントし、融合機の動作に必要なサービス層およびアプリ層のプロセスを所定の設定ファイルに従って起動している。従来の融合機では、起動されたサービス層およびアプリ層のプロセスが、各プロセスの処理の中で表示部、印刷部および撮像部などのハードウェア資源のチェックを行っていた。例えば特許文献2には、予め定義された動作定義情報に従いアプリを起動するプログラムの一例が記載されている。

【特許文献1】特開2002-84383号公報

【特許文献2】特開2000-20203号公報

【発明の開示】

【発明が解決しようとする課題】

【0005】

しかしながら、従来の融合機ではハードウェア資源を複数のプロセスで共通に利用しており、各プロセスの処理の中でハードウェア資源のチェックを行うと各プロセスに重複した部分が多くなるという問題があった。また、従来の融合機では各プロセスの処理の中でハードウェア資源のチェックを行うため、各プロセスを起動しなければハードウェア資源の有無、性能などのチェックを行うことができなかった。

【0006】

したがって、従来の融合機ではハードウェア資源が存在しない、ハードウェア資源の性能が低い等の理由で使用しないプロセスであっても、ハードウェア資源のチェックを行うために無駄に起動しなければならないという問題があった。なお、特許文献2はハードウェア資源のチェックを行うものでなく、上記の問題を解決することができない。

【0007】

本発明は、上記の点に鑑みなされたもので、各プログラムの重複部分を削減することができ、ハードウェア資源に関連するプログラムを効率良く起動することが可能な画像形成装置、プログラム起動方法及びプログラム起動プログラムを提供することを目的とする。

【課題を解決するための手段】

【0008】

そこで、上記課題を解決するため、本発明は、画像形成処理で使用されるハードウェア資源と、画像形成に係る処理を行うプログラムとを有する画像形成装置であって、前記ハードウェア資源に関するチェックを行うチェック処理と前記プログラムとの関連を設定する設定手段と、前記チェック処理を行い、そのチェック処理の結果に応じて前記チェック処理に関連する前記プログラムを起動する起動手段とを有することを特徴とする。

【0009】

また、本発明は、画像形成処理で使用されるハードウェア資源と、画像形成に係る処理を行うプログラムとを有する画像形成装置のプログラム起動方法であって、起動手段が、前記プログラムと前記ハードウェア資源に関するチェックを行うチェック処理との関連を設定した設定手段を解析する解析段階と、前記起動手段が、前記解析の結果に応じて前記チェック処理を実行するチェック処理実行段階と、前記起動手段が、前記チェック処理の結果に応じて前記チェック処理に関連する前記プログラムを起動するプログラム起動段階とを有することを特徴とする。

【0010】

また、本発明は、画像形成処理で使用されるハードウェア資源と、画像形成に係る処理を行うプログラムとを有するコンピュータを、前記ハードウェア資源に関するチェックを行うチェック処理と前記プログラムとの関連を設定する設定手段と、前記チェック処理を行い、そのチェック処理の結果に応じて前記チェック処理に関連する前記プログラムを起動する起動手段として機能させるためのプログラム起動プログラムであることを特徴とする。

【0011】

本発明によれば、プログラムを起動する起動手段がハードウェア資源に関するチェックを行うようにしたため、各プログラムの処理の中でハードウェア資源に関するチェックを行う必要がなくなり、各プログラムの重複部分を削減できる。

【0012】

また、本発明によれば、ハードウェア資源に関するチェックの結果に応じてプログラムを起動するため、使用しないプログラムを起動する必要がなくなり、プログラムを効率良く起動できる。

【発明の効果】**【0013】**

上述の如く、本発明によれば、各プログラムの重複部分を削減することができ、ハードウェア資源に関連するプログラムを効率良く起動することが可能な画像形成装置、プログラム起動方法及びプログラム起動プログラムを提供できる。

【発明を実施するための最良の形態】**【0014】**

次に、本発明を実施するための最良の形態を、以下の実施例に基づき図面を参照しつつ説明していく。

【0015】

図1は、本発明による融合機のソフトウェア構成について説明するための一実施例の構成図である。融合機1は、ソフトウェア群2と、融合機起動部3と、ハードウェア資源4とを含むように構成される。

【0016】

ハードウェア資源4は、プロッタ11と、スキャナ12と、ファクシミリなどのその他のハードウェアリソース13とを含む。ソフトウェア群2は、UNIX（登録商標）などのオペレーティングシステム（以下、OSという）上に起動されているアプリケーション層5とプラットフォーム6とを含む。

【0017】

アプリケーション層5は、プリンタ、コピー、ファックスおよびスキャナなどの画像形成にかかるユーザサービスにそれぞれ固有の処理を行うプログラムを含む。図1のアプリ

ケーション層 5 は、プリンタアプリ 21 と、コピーアプリ 22 と、ファックスアプリ 23 と、スキャナアプリ 24 と、ネットファイルアプリ 25 とを含む。なお、ネットファイルアプリ 25 はネットワークファイル用アプリケーションであり、融合機 1 にネットワークを介して接続されるネットワーク機器とのデータ通信を管理するものである。

【0018】

プラットフォーム 6 は、アプリケーション層 5 からの処理要求を解釈してハードウェア資源 4 の獲得要求を発生するコントロールサービス層 9 と、1 つ以上のハードウェア資源 4 の管理を行ってコントロールサービス層 9 からの獲得要求を調停するシステムリソースマネージャ（以下、SRM という）39 と、SRM 39 からの獲得要求に応じてハードウェア資源 4 の管理を行うハンドラ層 10 とを含む。

【0019】

コントロールサービス層 9 は、NCS 31, DCS 32, OCS 33, FCS 34, ECS 35, MCS 36, UCS 37, SCS 38 など、一つ以上のサービスモジュールを含む。なお、プラットフォーム 6 は予め定義されている関数により、アプリケーション層 5 からの処理要求を受信する API 53 を有するように構成されている。OS は、アプリケーション層 5 およびプラットフォーム 6 の各ソフトウェアをプロセスとして並列実行する。

【0020】

NCS（ネットワークコントロールサービス）31 のプロセスは、ネットワーク側から各プロトコルによって受信したデータを各アプリケーションに振り分ける際の仲介、又は各アプリケーションからのデータをネットワーク側に送信する際の仲介を行う。例えば NCS 31 は、融合機にネットワークを介して接続されるネットワーク機器とのデータ通信を制御する。

【0021】

DCS（デリバリーコントロールサービス）32 のプロセスは、融合機に蓄積されている文書データの配送などの制御を行う。OCS（操作パネルコントロールサービス）33 のプロセスは、後述する操作パネルの制御を行う。

【0022】

FCS（ファックスコントロールサービス）34 のプロセスは、アプリケーション層 5 から PSTN または ISDN 網を利用したファックスの送受信、バックアップ用のメモリで管理されている各種ファックスデータの登録又は引用、ファックスの読み取り、ファックスの受信印刷などを行うための API を提供する。

【0023】

ECS（エンジンコントロールサービス）35 のプロセスは、プロッタ 11, スキャナ 12, ハードウェアリソース 13 などのエンジン部の制御を行う。MCS（メモリコントロールサービス）36 のプロセスは、メモリの取得及び解放、HDD の利用、画像データの圧縮および伸張などの制御を行う。UCS（ユーザ情報コントロールサービス）37 のプロセスは、ユーザ情報の管理を行うものである。

【0024】

SCS（システムコントロールサービス）38 のプロセスは、操作部の制御、システム画面の表示、LED の表示、ハードウェア資源の管理、アプリケーションの管理、割り込みアプリケーションの制御などの処理を行う。

【0025】

SRM 39 のプロセスは、SCS 38 と共にシステムの制御およびハードウェア資源 4 の管理を行うものである。例えば SRM 39 のプロセスは、プロッタ 11 やスキャナ 12 などのハードウェア資源 4 を利用する上位層からの獲得要求に従って調停を行い、ハードウェア資源 4 の実行を制御する。

【0026】

具体的に、SRM 39 のプロセスは獲得要求されたハードウェア資源 4 が利用可能であるか（他の獲得要求により利用されていないか）を判定し、利用可能であれば獲得要求さ

れたハードウェア資源4が利用可能である旨を上位層に通知する。SRM39のプロセスは、上位層からの獲得要求に対してハードウェア資源4を利用するためのスケジューリングを行い、要求内容（プリンタエンジンによる紙搬送と作像動作，メモリの確保，ファイル生成など）を直接実施している。

【0027】

また、ハンドラ層10は後述するFCU（ファックスコントロールユニット）の管理を行うFCUH（ファックスコントロールユニットハンドラ）40と、プロセスに対するメモリの割り振り及びプロセスに割り振ったメモリの管理を行うIMH（イメージメモリハンドラ）41とを含む。SRM39及びFCUH40は、予め定義されている関数によりハードウェア資源4に対する処理要求を送信するエンジンI/F54を利用して、ハードウェア資源4に対する処理要求を行う。

【0028】

図1の構成により、融合機1は各アプリケーションで共通的に必要な処理をプラットフォーム6で一元的に処理することができる。次に、融合機1のハードウェア構成について説明する。

【0029】

図2は、本発明による融合機のハードウェア構成について説明するための一実施例の構成図である。図2の融合機1は、コントローラ60，操作パネル80，FCU81，エンジン部82を有する。

【0030】

コントローラ60は、CPU61，システムメモリ62，NB63，SB64，ASIC66，ローカルメモリ67，HDD68，NIC69，SDカード用スロット70，USB I/F71，IEEE1394 I/F72，セントロニクス I/F73を有する。

【0031】

操作パネル80は、コントローラ60のASIC66に接続されている。また、FCU81およびエンジン部82はコントローラ60のASIC66にPCIバス83を介して接続されている。

【0032】

コントローラ60は、ASIC66にローカルメモリ67，HDD68などが接続されると共に、CPU61とASIC66とがCPUチップセットのNB63を介して接続されている。なお、ASIC66とNB63とはAGP（Accelerated Graphics Port）65を介して接続されている。

【0033】

CPU61は、融合機1の全体制御を行うものである。図1の融合機1では、CPU61がコントロールサービス層9を形成する1つ以上のサービスモジュールと、SRM39と、ハンドラ層10を形成するFCUH40，IMH41とをOS上に起動させた後、アプリケーション層5を形成するプリンタアプリ21，コピーアプリ22，ファックスアプリ23，スキャナアプリ24，ネットファイルアプリ25を起動して実行させる。

【0034】

NB（ノースブリッジ）63は、CPU61，システムメモリ62，SB64，ASIC66，NIC69，SDカード用スロット70，USB I/F71，IEEE1394 I/F72及びセントロニクス I/F73を接続するためのブリッジである。NB63は、SB64，NIC69，SDカード用スロット70，USB I/F71，IEEE1394 I/F72及びセントロニクス I/F73とPCIバス74を介して接続されている。なお、SB（サウスブリッジ）64は、PCIバス74とROMや周辺デバイス等とを接続するためのブリッジである。

【0035】

システムメモリ62は、描画用メモリ等として用いるメモリである。ローカルメモリ67は、コピー用画像バッファ，符号バッファ等として用いるメモリである。ASIC66

は、画像処理用のハードウェア要素を有する画像処理用途向けのICである。また、HDD 68は画像データの蓄積、文書データの蓄積、プログラムの蓄積、フォントデータの蓄積、フォームの蓄積などを行うストレージ（補助記憶装置）の一例である。

【0036】

NIC（ネットワークインターフェースカード）69は、融合機1をインターネットやLAN等のネットワークに接続するインターフェース機器である。SDカード用スロット70はSDカードを挿抜可能なものであり、SDカードの挿入または抜き出しに応じた割り込みをデバイスドライバに対して行う。

【0037】

USB I/F 70、IEEE1394 I/F 71およびセントロニクス I/F 72は、夫々の規格に準じたインターフェースである。操作パネル80は、操作者からの入力操作を受け付けると共に、操作者に向けた表示を行う操作部である。なお、FCU 81はバックアップ用のメモリを有している。FCU 81が有するメモリは、例えば融合機1の電源がOFFのときに受信したファクシミリデータを一時的に格納するために利用される。

【0038】

図3は、融合機起動部の一例の構成図である。図1の融合機起動部3は、融合機1の電源投入時に最初に実行され、アプリケーション層5やプラットフォーム6を起動するものである。融合機起動部3は、ROMモニタ51と、プログラム起動部52とを含む。融合機起動部3の処理について図4のフローチャートを参照しつつ説明する。

【0039】

図4は、融合機起動部の処理の一例のフローチャートである。ステップS101では、融合機1の電源ONにより、BIOSおよびブートローダとしてのROMモニタ51が実行される。ROMモニタ51は、ハードウェアの初期化、コントローラ60の診断、ソフトウェアの初期化などを行う。ステップS101に続いてステップS102に進み、ROMモニタ51は、OSおよびルートファイルシステムをシステムメモリ62上に展開してOSを起動する。そして、OSはルートファイルシステムをマウントする。

【0040】

ステップS102に続いてステップS103に進み、OSは起動時に接続されたデバイスのデバイス情報（例えば、CPU 61のクロック周波数、システムメモリ62およびローカルメモリ57のメモリサイズ、コントローラ60のボードタイプなど）を取得する。

【0041】

ステップS103に続いてステップS104に進み、OSはアプリ／サービス起動プログラムとしてのプログラム起動部52を起動する。プログラム起動部52はシステムメモリ62およびローカルメモリ67上にメモリ領域を確保する。プログラム起動部52は、融合機1で最初に起動されるプロセスである。ステップS104に続いてステップS105に進み、プログラム起動部52は設定ファイルに従ってファイルシステムをマウントする。

【0042】

また、プログラム起動部52は設定ファイルに従って、ハードウェア資源に関するチェックを行うチェック処理を実行する。プログラム起動部52は、チェック処理が正常に終了したか否かで、設定ファイルに記述されたアプリケーション層5およびプラットフォーム6のプログラム（以下、本体プログラムという）を起動するか否かを判定する。

【0043】

本体プログラムを起動すると判定した場合、プログラム起動部52は、その本体プログラムを設定ファイルに従ってROMなどから読み出し、読み出した本体プログラムをシステムメモリ62、ローカルメモリ67上に確保したメモリ領域に展開して、アプリケーション層5およびプラットフォーム6のプロセスを起動する。以下、プログラム起動部52が行うステップS105の処理について、詳細に説明していく。

【実施例1】

【0044】

図5は、プログラム起動部の処理の一例のフローチャートである。まず、ステップS110では、プログラム起動部52が設定ファイルを解析する。ステップS110に続いてステップS111に進み、プログラム起動部52は設定ファイルに従ってファイルシステムをマウントする。

【0045】

ステップS111に続いてステップS112に進み、プログラム起動部52は設定ファイルに記述されたexecコマンドを1つ読み出し、そのexecコマンドに「-h」オプションがあるか否かを判定する。例えば図6のような設定ファイルの場合、プログラム起動部52は1行目のexecコマンドに「-h」オプションがあると判定する。execコマンドに「-h」オプションがあると判定すると（S112においてYES）、プログラム起動部52はステップS113に進み、execコマンドで指定されたチェック処理を実行する。例えば図6のような設定ファイルの場合、プログラム起動部52は1行目のexecコマンドで指定されたチェック処理「fcucheck」を実行する。ステップS113のチェック処理では、ハードウェア資源に関するチェック（例えばハードウェア資源の有無、性能などのチェック）を行い、チェックの結果を取得する。

【0046】

ステップS113に続いてステップS114に進み、プログラム起動部52は実行したチェック処理の結果に基づいて、チェック処理が正常終了したか異常終了したかを判定する。チェック処理が正常終了したと判定すると（S114においてYES）、プログラム起動部52はステップS115に進む。

【0047】

ステップS115では、プログラム起動部52が、execコマンドで指定された本体プログラムを起動する。例えば図6のような設定ファイルの場合、プログラム起動部52は1行目のexecコマンドで指定された本体プログラム「/fax/bin/fax」を起動する。この後、プログラム起動部52はチェック処理を終了する。

【0048】

ステップS115に続いてステップS116に進み、プログラム起動部52は起動すべき本体プログラム、言い換えれば読み込んでいないexecコマンドが設定ファイルにまだ存在するか否かを判定する。まだ読み込んでいないexecコマンドが設定ファイルに存在すると判定すると（S116においてYES）、プログラム起動部52はステップS112に戻り、まだ読み込んでいないexecコマンドを設定ファイルから読み込んでステップS112以降の処理を行う。

【0049】

一方、まだ読み込んでいないexecコマンドが設定ファイルに存在しないと判定すると（S116においてNO）、プログラム起動部52は処理を終了する。なお、ステップS112において、execコマンドに「-h」オプションがないと判定すると（S112においてNO）、プログラム起動部52はステップS115に進み、execコマンドで指定された本体プログラムを起動する。即ち、execコマンドに「-h」オプションがなければ、プログラム起動部52はチェック処理を実行することなくexecコマンドで指定された本体プログラムを起動する。

【0050】

また、ステップS114において、チェック処理が正常終了していないと判定すると（S114においてNO）、プログラム起動部52はステップS116に進む。即ち、チェック処理が異常終了した場合、プログラム起動部52はexecコマンドで指定された本体プログラムを起動しない。

【0051】

図5のフローチャートの処理により、プログラム起動部52はチェック処理が正常終了したときにexecコマンドで指定された本体プログラムを起動する一方、チェック処理が異常終了したときにexecコマンドで指定された本体プログラムを起動しないことに

より融合機1の動作を抑制できる。

【0052】

なお、図5のフローチャートでは、コマンドの一例としてexecコマンドを例に説明したが、mountコマンドであってもよい。この場合、プログラム起動部52はチェック処理が正常終了したときにmountコマンドで指定されたマウントを行う一方、チェック処理が異常終了したときにmountコマンドで指定されたマウントを行わないことにより融合機1のマウント動作を抑制できる。

【0053】

次に、図6の設定ファイルを用いて複数のチェック処理について説明していく。図6の設定ファイルでは、1行目のexecコマンドに「-h」オプションがあるので、チェック処理「fcuccheck」が実行される。例えばプログラム起動部52は、図7のようなチェック処理「fcuccheck」を行う。

【0054】

図7は、チェック処理「fcuccheck」の一例のフローチャートである。ステップS120では、プログラム起動部52がFCU81のデバイスドライバをオープンする為の処理を行う。ステップS120に続いてステップS121に進み、プログラム起動部52はステップS120でのデバイスドライバのオープンが成功したか否かを判定する。

【0055】

デバイスドライバのオープンが成功したと判定すると（S121においてYES）、プログラム起動部52はステップS123に進み、FCU81が融合機1に接続されているとみなして、正常終了を表すチェック結果を取得する。一方、デバイスドライバのオープンが失敗したと判定すると（S121においてNO）、プログラム起動部52はステップS122に進み、FCU81が既にビジー状態であるか否かを判定する。なお、FCU81がビジー状態であるか否かは、例えば「errno」に「EBUSY」が入っているか否かで確認できる。

【0056】

FCU81が既にビジー状態であると判定すると（S122においてYES）、プログラム起動部52はステップS124に進み、FCU81が融合機1に接続されているとみなして、正常終了を表すチェック結果を取得する。一方、FCU81が既にビジー状態であると判定しなかった場合（S122においてNO）、プログラム起動部52はステップS125に進み、FCU81が融合機1に接続されていないとみなして、異常終了を表すチェック結果を取得する。

【0057】

図7のフローチャートの処理により、プログラム起動部52はFCU81が融合機1に接続されている場合に正常終了を表すチェック結果を取得し、FCU81が融合機1に接続されていない場合に異常終了を表すチェック結果を取得できる。なお、プログラム起動部52は正常終了を表すチェック結果を取得したときにアプリ「fax」を起動し、異常終了を表すチェック結果を取得したときにアプリ「fax」を起動しない。

【0058】

したがって、プログラム起動部52は正常終了または異常終了を表すチェック結果を利用して、FCU81が融合機1に接続されている場合にアプリ「fax」を起動し、接続されていない場合にアプリ「fax」を起動しないようにすることができる。即ち、プログラム起動部52はアプリ「fax」の起動をFCU81の接続状態によって抑制できる。なお、図7のフローチャートでは、デバイスの一例としてのFCU81を用いて説明したが、如何なるデバイスであってもよい。つまり、図7のフローチャートによれば、オプションボード等のデバイスの接続状態によってアプリの起動を抑制することができる。

【0059】

図6の設定ファイルでは、1行目のexecコマンドの処理が終了すると、2行目のexecコマンドの処理が行われる。図6の設定ファイルでは、2行目のexecコマンドに「-h」オプションがあるので、チェック処理「cpuccheck1」が実行される。

例えばプログラム起動部52は、図8のようなチェック処理「cpucheck1」を行う。

【0060】

図8は、チェック処理「cpucheck1」の一例のフローチャートである。ステップS130では、プログラム起動部52がシステムコール「getINFO(CPU)」を呼び、デバイス情報に含まれているCPU61のクロック周波数をOSから取得する。

【0061】

ステップS130に続いてステップS131に進み、プログラム起動部52はステップS130で取得したCPU61のクロック周波数が500MHz以下であるか否かを判定する。CPU61のクロック周波数が500MHz以下であると判定すると（S131においてYES）、プログラム起動部52はステップS132に進み、正常終了を表すチェック結果を取得する。一方、CPU61のクロック周波数が500MHz以下でないと判定すると（S131においてNO）、プログラム起動部52はステップS133に進み、異常終了を表すチェック結果を取得する。

【0062】

図8のフローチャートの処理により、プログラム起動部52はCPU61のクロック周波数が500MHz以下である場合に正常終了を表すチェック結果を取得し、CPU61のクロック周波数が500MHz以下でない場合に異常終了を表すチェック結果を取得できる。なお、プログラム起動部52は正常終了を表すチェック結果を取得したときにアプリ「setfont_bitmap」を起動し、異常終了を表すチェック結果を取得したときにアプリ「setfont_bitmap」を起動しない。

【0063】

したがって、プログラム起動部52は正常終了または異常終了を表すチェック結果を利用して、CPU61のクロック周波数が500MHz以下である場合にプリンタが使用するデフォルトフォントをビットマップ形式に設定する。即ち、プログラム起動部52はCPU61のクロック周波数が500MHz以下であっても、一定時間内にプリントできるようなデフォルトフォントに変えることで、プリントの高速性を優先できる。

【0064】

図6の設定ファイルでは、2行目のexecコマンドの処理が終了すると、3行目のexecコマンドの処理が行われる。図6の設定ファイルでは、3行目のexecコマンドに「-h」オプションがあるので、チェック処理「cpucheck2」が起動される。例えばプログラム起動部52は、図9のようなチェック処理「cpucheck2」を行う。

【0065】

図9は、チェック処理「cpucheck2」の一例のフローチャートである。ステップS140では、プログラム起動部52がシステムコール「getINFO(CPU)」を呼び、デバイス情報に含まれているCPU61のクロック周波数をOSから取得する。

【0066】

ステップS140に続いてステップS141に進み、プログラム起動部52はステップS140で取得したCPU61のクロック周波数が501MHz以上であるか否かを判定する。CPU61のクロック周波数が501MHz以上であると判定すると（S141においてYES）、プログラム起動部52はステップS142に進み、正常終了を表すチェック結果を取得する。一方、CPU61のクロック周波数が501MHz以上でないと判定すると（S141においてNO）、プログラム起動部52はステップS143に進み、異常終了を表すチェック結果を取得する。

【0067】

図9のフローチャートの処理により、プログラム起動部52はCPU61のクロック周波数が501MHz以上である場合に正常終了を表すチェック結果を取得し、CPU61のクロック周波数が501MHz以上でない場合に異常終了を表すチェック結果を取得できる。なお、プログラム起動部52は、正常終了を表すチェック結果を取得したときにア

プリ「setfont__vector」を起動し、異常終了を表すチェック結果を取得したときにアプリ「setfont__vector」を起動しない。

【0068】

したがって、プログラム起動部52は正常終了または異常終了を表すチェック結果を利用して、CPU61のクロック周波数が501MHz以上である場合にプリンタが使用するデフォルトフォントをベクトル画像に設定する。即ち、プログラム起動部52はCPU61のクロック周波数が501MHz以上であるときに、精細なデフォルトフォントに変えることで、プリントの高画質化が可能である。

【0069】

図8および図9のフローチャートの処理により、本発明の融合機1は一定の時間内にプリントできるように、CPU61の性能に応じて用いるデフォルトフォントを変えることで、プリントの高速性を優先することができる。

【0070】

図6の設定ファイルでは、3行目のexecコマンドの処理が終了すると、4行目のexecコマンドの処理が行われる。図6の設定ファイルでは、4行目のexecコマンドに「-h」オプションがあるので、チェック処理「memcheck1」が実行される。例えばプログラム起動部52は、図10のようなチェック処理「memcheck1」を行う。

【0071】

図10は、チェック処理「memcheck1」の一例のフローチャートである。ステップS150では、プログラム起動部52がシステムコール「getINFO(mem)」を呼び、デバイス情報に含まれるシステムメモリ62及びローカルメモリ67のメモリサイズをOSから取得する。ステップS150に続いてステップS151に進み、プログラム起動部52はステップS150で取得したメモリサイズが64MB以上128MB以下であるか否かを判定する。

【0072】

メモリサイズが64MB以上128MB以下であると判定すると（S151においてYES）、プログラム起動部52はステップS152に進み、正常終了を表すチェック結果を取得する。一方、メモリサイズが64MB以上128MB以下でないと判定すると（S151においてNO）、プログラム起動部52はステップS153に進み、異常終了を表すチェック結果を取得する。

【0073】

図10のフローチャートの処理により、プログラム起動部52はメモリサイズが64MB以上128MB以下である場合に正常終了を表すチェック結果を取得し、メモリサイズが64MB以上128MB以下でない場合に異常終了を表すチェック結果を取得できる。なお、プログラム起動部52は正常終了を表すチェック結果を取得したときにhttpのデーモン（以下、httpdという）を5個起動し、異常終了を表すチェック結果を取得したときにhttpdを起動しない。

【0074】

したがって、プログラム起動部52は正常終了または異常終了を表すチェック結果を利用することで、システムメモリ62及びローカルメモリ67のメモリサイズが小さい場合であっても、起動するhttpdの数を少なくして、一つの要求に対する処理時間の増大を防ぐことができる。

【0075】

図6の設定ファイルでは、4行目のexecコマンドの処理が終了すると、5行目のexecコマンドの処理が行われる。図6の設定ファイルでは、5行目のexecコマンドに「-h」オプションがあるので、チェック処理「memcheck2」が実行される。例えばプログラム起動部52は、図11のようなチェック処理「memcheck2」を行う。

【0076】

図11は、チェック処理「memcheck2」の一例のフローチャートである。ステップS160では、プログラム起動部52がシステムコール「getINFO(mem)」を呼び、デバイス情報に含まれるシステムメモリ62及びローカルメモリ67のメモリサイズをOSから取得する。ステップS160に続いてステップS161に進み、プログラム起動部52はステップS160で取得したメモリサイズが128MB以上であるか否かを判定する。

【0077】

メモリサイズが128MB以上であると判定すると（S161においてYES）、プログラム起動部52はステップS162に進み、正常終了を表すチェック結果を取得する。一方、メモリサイズが128MB以上でないと判定すると（S161においてNO）、プログラム起動部52はステップS163に進み、異常終了を表すチェック結果を取得する。

【0078】

図11のフローチャートの処理により、プログラム起動部52はメモリサイズが128MB以上である場合に正常終了を表すチェック結果を取得し、メモリサイズが128MB以上でない場合に異常終了を表すチェック結果を取得できる。なお、プログラム起動部52は正常終了を表すチェック結果を取得したときにhttpdを10個起動し、異常終了を表すチェック結果を取得したときにhttpdを起動しない。

【0079】

したがって、プログラム起動部52は正常終了または異常終了を表すチェック結果を利用することで、システムメモリ62及びローカルメモリ67のメモリサイズが大きい場合に、起動するhttpdの数を多くして、クライアントからの要求にすぐ対応することができる。

【0080】

図10および図11のフローチャートの処理により、本発明の融合機1はシステムメモリ62及びローカルメモリ67のメモリサイズに応じて、起動するhttpdの数を適切に変えることができる。

【0081】

次に、図12の設定ファイルを用いてプログラム起動部52のチェック処理について説明する。図12の設定ファイルでは、mountコマンドに「-h」オプションがあるので、チェック処理「hddnonexist」が実行される。例えばプログラム起動部52は、図13のようなチェック処理「hddnonexist」を行う。

【0082】

図13は、チェック処理「hddnonexist」の一例のフローチャートである。ステップS170では、プログラム起動部52がシステムコール「getINFO(hdd)」を呼び、デバイス情報に含まれているHDD接続有無情報をOSから取得する。

【0083】

ステップS170に続いてステップS171に進み、プログラム起動部52はステップS171で取得したHDD接続有無情報に応じてHDDが融合機1に接続されているか否かを判定する。HDDが融合機1に接続されていないと判定すると（S171においてNO）、プログラム起動部52はステップS172に進み、正常終了を表すチェック結果を取得する。一方、HDDが融合機1に接続されていると判定すると（S171においてYES）、プログラム起動部52はステップS173に進み、異常終了を表すチェック結果を取得する。

【0084】

図13のフローチャートの処理により、プログラム起動部52はHDDが融合機1に接続されていない場合に正常終了を表すチェック結果を取得し、HDDが融合機1に接続されている場合に異常終了を表すチェック結果を取得できる。なお、プログラム起動部52は、正常終了を表すチェック結果を取得したときにramdiskをマウントする。具体的には、「/dev/md0c」がマウントポイント「/ramdisk」へマウントさ

れる。また、プログラム起動部 52 は異常終了を表すチェック結果を取得したときに `ramdisk` をマウントしない。

【0085】

したがって、プログラム起動部 52 は正常終了または異常終了を表すチェック結果を利用することで、HDD が融合機 1 に接続されていない場合に `ramdisk` をマウントする。即ち、プログラム起動部 52 は HDD が融合機 1 に接続されていない場合であっても、PDL ストレージ用のローカルストレージ用デバイスとして `ramdisk` を用いることができる。なお、プログラム起動部 52 は HDD が融合機 1 に接続されている場合、PDL ストレージ用のローカルストレージ用デバイスとして HDD を用いることができる。

【0086】

次に、図 14 の設定ファイルを用いてプログラム起動部 52 のチェック処理について説明する。図 14 の設定ファイルは、図 15 のように SD カード内に格納されている。図 15 に表された SD カードの場合、「`xxx.cnf`」が設定ファイル、「`module/xxx.mod`」がマウント及び実行対象のモジュールファイルを表している。

【0087】

図 14 の設定ファイルでは、`mount` コマンドに「`-h`」オプションがあるので、チェック処理「`sd command`」が実行される。例えばプログラム起動部 52 は、図 16 のようなチェック処理「`sd command`」を行う。

【0088】

図 16 は、チェック処理「`sd command`」の一例のフローチャートである。ステップ S180 では、プログラム起動部 52 が SD カード内の図 14 のような設定ファイルを解析する。ステップ S180 に続いてステップ S181 に進み、プログラム起動部 52 はステップ S180 の解析結果から設定ファイルに SD コマンドが存在するか否かを判定する。SD コマンドが存在すると判定すると（S181 において YES）、プログラム起動部 52 はステップ S182 に進む。一方、SD コマンドが存在しないと判定すると（S181 において NO）、プログラム起動部 52 はステップ S183 に進む。

【0089】

ステップ S182 では、プログラム起動部 52 が、SD コマンドで指定されているスロットと SD カードが挿入されているスロットとが一致しているかを判定する。例えば図 14 のような設定ファイルの場合、SD コマンドで指定されているスロットが「2」であるため、SD カードが挿入されているスロットが「2」であるときに、プログラム起動部 52 は SD コマンドで指定されているスロットと SD カードが挿入されているスロットとが一致していると判定する。

【0090】

SD コマンドで指定されているスロットと SD カードが挿入されているスロットとが一致していると判定すると（S182 において YES）、プログラム起動部 52 はステップ S183 に進み、正常終了を表すチェック結果を取得する。一方、SD コマンドで指定されているスロットと SD カードが挿入されているスロットとが一致していないと判定すると（S182 において NO）、プログラム起動部 52 はステップ S184 に進み、異常終了を表すチェック結果を取得する。

【0091】

図 16 のフローチャートの処理により、プログラム起動部 52 は SD コマンドで指定されているスロットと SD カードが挿入されているスロットとが一致している場合に正常終了を表すチェック結果を取得し、SD コマンドで指定されているスロットと SD カードが挿入されているスロットとが一致していない場合に異常終了を表すチェック結果を取得できる。

【0092】

なお、プログラム起動部 52 は正常終了を表すチェック結果を取得したときに `gzip` 圧縮された ROMFS 形式のモジュールファイル「`xxx.mod`」をマウントポイント「`/mnt`」へマウントし、そのモジュールファイルを実行する。また、プログラム起動

部52は異常終了を表すチェック結果を取得したときにモジュールファイルをマウント及び実行しない。

【0093】

したがって、プログラム起動部52は正常終了または異常終了を表すチェック結果を利用することで、SDコマンドで指定されているスロットと異なるスロットに挿入されているSDカード内のモジュールファイルのマウント及び実行を抑制することができる。

【0094】

図6の設定ファイルの場合、本体プログラムと、チェック処理と、プログラム起動部52と、OSと、ハードウェア資源との関係は、例えば図17のようになる。図17は、本体プログラムと、チェック処理と、プログラム起動部52と、OSと、ハードウェアとの関係を表した一例の関係図である。

【0095】

プログラム起動部52はチェック処理を順番に実行し、チェック処理が正常終了したとき、そのチェック処理に対応する本体プログラムを起動する。図17では、各チェック処理の上に位置している本体プログラムがチェック処理に対応する本体プログラムである。

【実施例2】

【0096】

図17は、プログラム起動部52のチェック処理と本体プログラムとを1対1に対応させた例を表しているが、図18のようにチェック処理と本体プログラムとを1対nに対応させるようにしてもよい。

【0097】

図18は、チェック処理と本体プログラムとが1対nに対応する例について説明するための図である。図18の例では、プログラム起動部52のチェック処理aと本体プログラムa、bとが対応している。プログラム起動部52はチェック処理aを実行し、チェック処理aが正常終了したとき、そのチェック処理aに対応する本体プログラムa、bを起動する。図18のようにチェック処理aと本体プログラムa、bとが対応している場合、設定ファイルは図19のように構成される。図19は、設定ファイルの他の一例の構成図である。なお、プログラム起動部52の処理は図5と同様であるので説明を省略する。

【0098】

また、同じチェック処理によるチェックの結果に基づき、ディレクトリ下の複数の本体プログラムを起動するような場合、チェックの結果に基づいてディレクトリのマウントを抑制すると効率的である。

【0099】

図20は、マウントを抑制する設定ファイルの一例の構成図である。図20のような設定ファイルの場合、プログラム起動部52は1行目のmountコマンドで指定されたチェック処理「memcheck3」を起動する。例えば図20のチェック処理「memcheck3」は、システムメモリ62及びローカルメモリ67のメモリサイズが64MB以上のときに正常終了するものとする。

【0100】

プログラム起動部52は、チェック処理が正常終了するとmountコマンドで指定された「web.romfs」をディレクトリ「/web」へマウントする一方、チェック処理が異常終了するとマウントしない。例えば図20の設定ファイルの記述の下に図6のような記述があれば、ディレクトリ「/web」以下のプログラム「/web/bin/httpd」なども起動されなくなる。したがって、プログラム起動部52はシステムメモリ62及びローカルメモリ67のメモリサイズに応じて無駄にメモリを消費することを避けることができる。

【0101】

なお、図20の設定ファイルによるマウント処理は、図5のステップS11で行われる処理である。また、図20の設定ファイルでは、メモリサイズに応じてディレクトリのマウントを抑制する例について表しているが、ハードウェア資源の有無やCPU性能などに

応じてディレクトリのマウントを抑制するようにしてもよい。

【実施例 3】

【0102】

図 18 は、プログラム起動部 52 のチェック処理と本体プログラムとを 1 対 n に対応させた例を表しているが、図 21 のようにチェック処理と本体プログラムとを n 対 1 に対応させるようにしてもよい。

【0103】

図 21 は、チェック処理と本体プログラムとが n 対 1 に対応する例について説明するための図である。図 21 の例では、プログラム起動部 52 のチェック処理 a 、 b と本体プログラム a とが対応している。プログラム起動部 52 はチェック処理 a 、 b を起動し、チェック処理 a 、 b が正常終了したとき、チェック処理 a 、 b に対応している本体プログラム a を起動する。図 21 のようにチェック処理 a 、 b と本体プログラム a とが対応している場合、設定ファイルは図 22 のように構成される。図 22 は、設定ファイルの他の一例の構成図である。

【0104】

図 22 の設定ファイルでは、1 つの `exec` コマンドに「`-h`」オプションのある 2 つのチェック処理「チェック処理 a 」及び「チェック処理 b 」が設定されている。したがって、プログラム起動部 52 は図 5 のステップ S113 でチェック処理 a 、 b を実行する。そして、プログラム起動部 52 はステップ S114 でチェック処理 a 、 b が両方とも正常終了したか否かを判定し、両方とも正常終了したと判定したときに `exec` コマンドで指定された本体プログラム a を起動する。なお、プログラム起動部 52 のステップ S113 及び S114 以外の処理は図 5 と同様であるので説明を省略する。

【実施例 4】

【0105】

融合機 1 では、設定ファイルに応じて、同一のチェック処理が繰り返し実行されることもある。したがって、以前に実行したチェック処理を再び実行する場合は、そのチェック処理のチェックの結果を利用する方が処理時間の短縮の観点から望ましい。そこで、本発明の融合機 1 は図 23 及び図 24 のような処理を行うことにより、以前に起動されたチェック処理のチェックの結果を利用できるようにする。

【0106】

図 23 及び図 24 は、プログラム起動部が行うチェック処理の一例のフローチャートである。なお、ステップ S190～S193 までの処理は図 5 のステップ S110～S113 の処理と同様であり、説明を省略する。

【0107】

ステップ S194 では、プログラム起動部 52 が、所定のメモリ領域にチェックの結果が格納されているか否かで既にチェック済みであるか否かを判定する。例えばチェックの結果を格納する所定のメモリ領域には、OS を介することなくプログラム起動部 52 のプロセスが直接アクセス可能なメモリ上の領域を利用できる。チェック済みであると判定すると（S194 において YES）、プログラム起動部 52 はステップ S195 に進み、所定のメモリ領域からチェックの結果を読み出したあとステップ S198 に進む。一方、チェック済みでないと判定すると（S194 において NO）、プログラム起動部 52 はステップ S196 に進み、前述したようなハードウェア資源に関するチェックを行う。

【0108】

ステップ S196 に続いてステップ S197 に進み、プログラム起動部 52 はチェックの結果を所定のメモリ領域に書き込んだあとステップ S198 に進む。なお、ステップ S198～S200 の処理は、図 5 のステップ S114～S116 の処理と同様であり、説明を省略する。図 23 および図 24 のフローチャートにより、以前に起動されたチェック処理のチェックの結果を利用できる。

【0109】

さらに、融合機 1 は起動時に全てのチェック処理を実行し、全てのチェック処理による

チェックの結果を所定のメモリ領域に予め書き込んでおいてもよい。この場合、融合機 1 は起動直後にチェック処理によるチェックを行ったあと、所定のメモリ領域に書き込まれたチェックの結果を利用することができるので、処理時間の短縮が可能となる。

【0110】

本発明は、具体的に開示された実施例に限定されるものではなく、特許請求の範囲から逸脱することなく、種々の変形や変更が可能である。

【図面の簡単な説明】

【0111】

【図1】本発明による融合機のソフトウェア構成について説明するための一実施例の構成図である。

【図2】本発明による融合機のハードウェア構成について説明するための一実施例の構成図である。

【図3】融合機起動部の一例の構成図である。

【図4】融合機起動部の処理の一例のフローチャートである。

【図5】プログラム起動部の処理の一例のフローチャートである。

【図6】設定ファイルの一例の構成図である。

【図7】チェック処理「fcuccheck」の一例のフローチャートである。

【図8】チェック処理「cpuccheck1」の一例のフローチャートである。

【図9】チェック処理「cpuccheck2」の一例のフローチャートである。

【図10】チェック処理「memcheck1」の一例のフローチャートである。

【図11】チェック処理「memcheck2」の一例のフローチャートである。

【図12】設定ファイルの他の一例の構成図である。

【図13】チェック処理「hddnonexist」の一例のフローチャートである。

。

【図14】設定ファイルの他の一例の構成図である。

【図15】SDカード内に格納されているファイルの一例のイメージ図である。

【図16】チェック処理「sdcommand」の一例のフローチャートである。

【図17】本体プログラムと、チェック処理と、プログラム起動部と、OSと、ハードウェアとの関係を表した一例の関係図である。

【図18】チェック処理と本体プログラムとが1対nに対応する例について説明するための図である。

【図19】設定ファイルの他の一例の構成図である。

【図20】マウントを抑制する設定ファイルの一例の構成図である。

【図21】チェック処理と本体プログラムとがn対1に対応する例について説明するための図である。

【図22】設定ファイルの他の一例の構成図である。

【図23】プログラム起動部が行うチェック処理の一例のフローチャートである（1／2）。

【図24】プログラム起動部が行うチェック処理の一例のフローチャートである（2／2）。

【符号の説明】

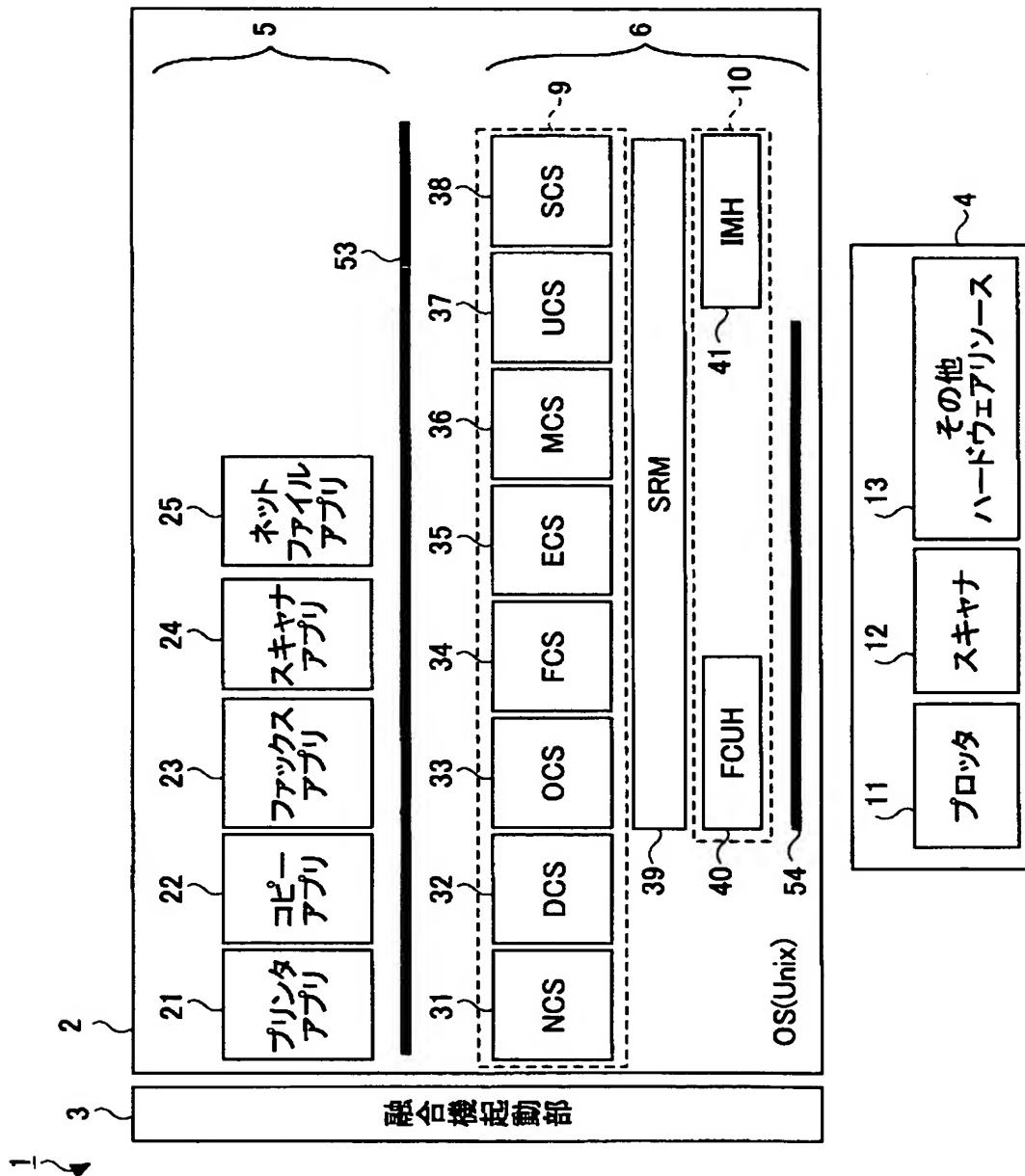
【0112】

- 1 融合機
- 2 ソフトウェア群
- 3 融合機起動部
- 4 ハードウェア資源
- 5 アプリケーション層
- 6 プラットフォーム
- 9 コントロールサービス層
- 10 ハンドラ層

1 1 プロッタ
1 2 スキャナ
1 3 ハードウェアリソース
2 1 プリンタアプリ
2 2 コピーアプリ
2 3 ファックスアプリ
2 4 スキャナアプリ
2 5 ネットファイルアプリ
3 1 ネットワークコントロールサービス (NCS)
3 2 デリバリーコントロールサービス (DCS)
3 3 オペレーションパネルコントロールサービス (OCS)
3 4 ファックスコントロールサービス (FCS)
3 5 エンジンコントロールサービス (ECS)
3 6 メモリコントロールサービス (MCS)
3 7 ユーザインフォメーションコントロールサービス (UCS)
3 8 システムコントロールサービス (SCS)
3 9 システムリソースマネージャ (SRM)
4 0 ファックスコントロールユニットハンドラ (FCUH)
4 1 イメージメモリハンドラ (IMH)
5 1 ROMモニタ
5 2 プログラム起動部
5 3 アプリケーションプログラムインターフェース (API)
5 4 エンジン I / F
6 0 コントローラ
6 1 CPU
6 2 システムメモリ
6 3 ノースブリッジ (NB)
6 4 サウスブリッジ (SB)
6 5 AGP (Accelerated Graphics Port)
6 6 ASIC
6 7 ローカルメモリ
6 8 ハードディスク装置 (HDD)
6 9 ネットワークインターフェースコントローラ (NIC)
7 0 SDカード用スロット
7 1 USB I / F
7 2 IEEE 1394 I / F
7 3 セントロニクス I / F
8 0 操作パネル
8 1 ファックスコントロールユニット (FCU)
8 2 エンジン部
8 3 PCIバス

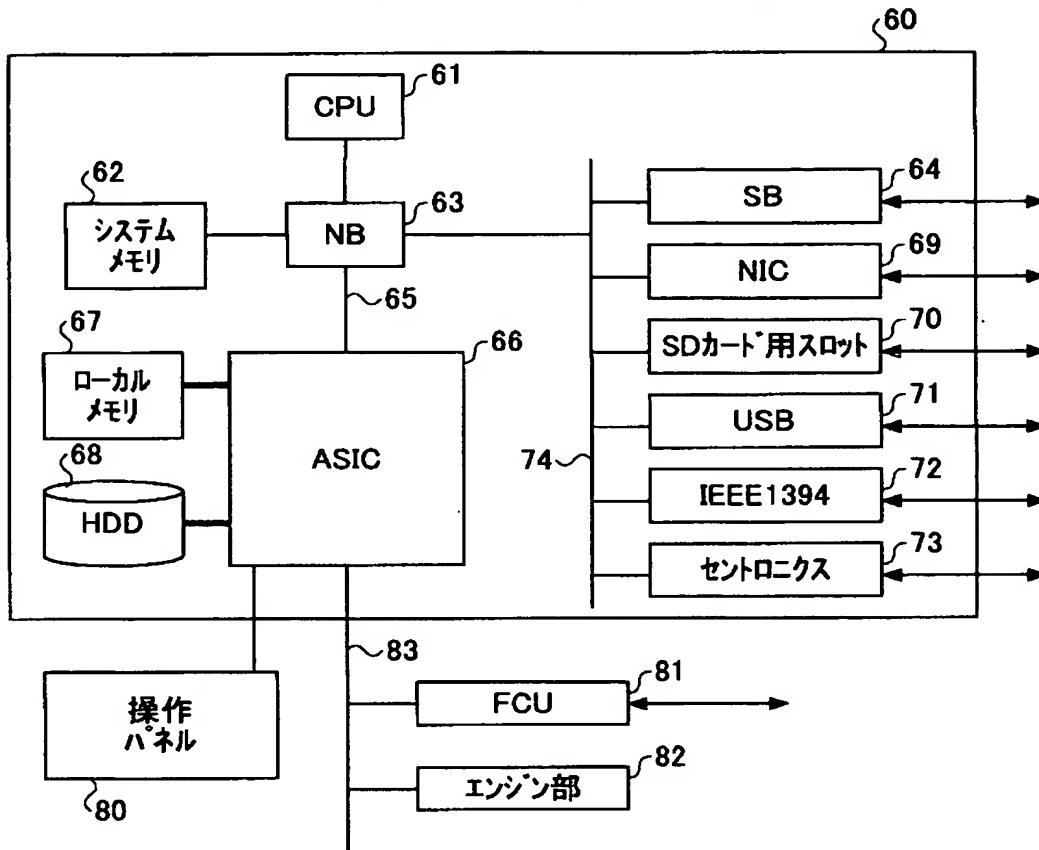
【書類名】 図面
【図 1】

本発明による融合機のソフトウェア構成について説明するための一実施例の構成図



【図 2】

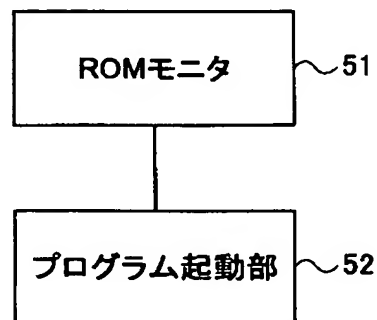
本発明による融合機のハードウェア構成について
説明するための一実施例の構成図



【図 3】

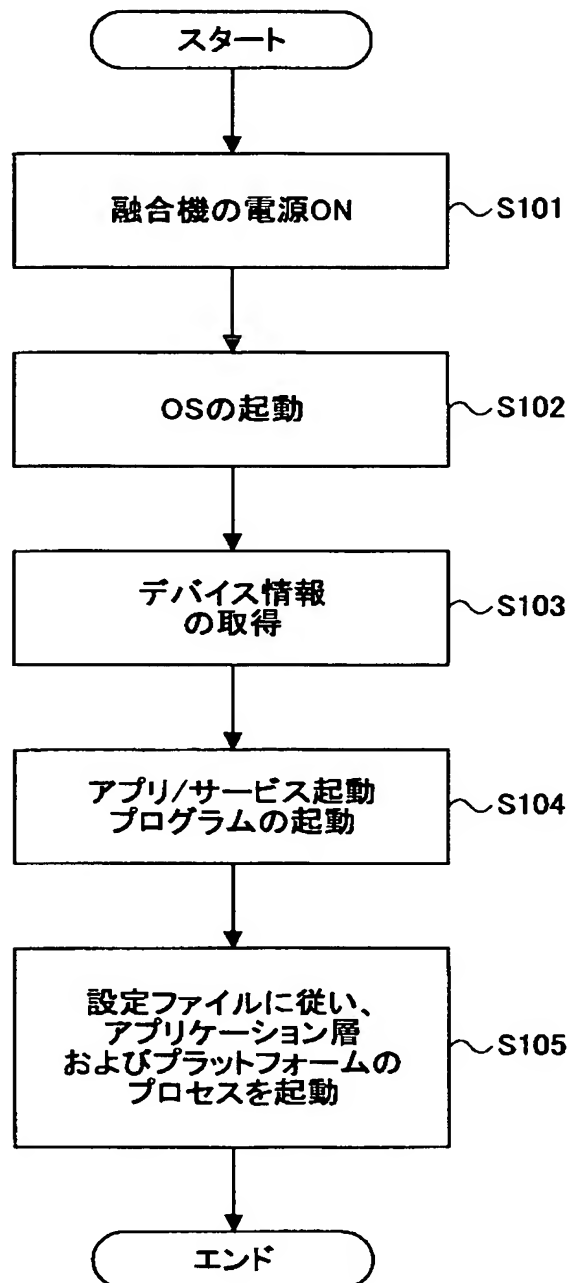
融合機起動部の一例の構成図

3



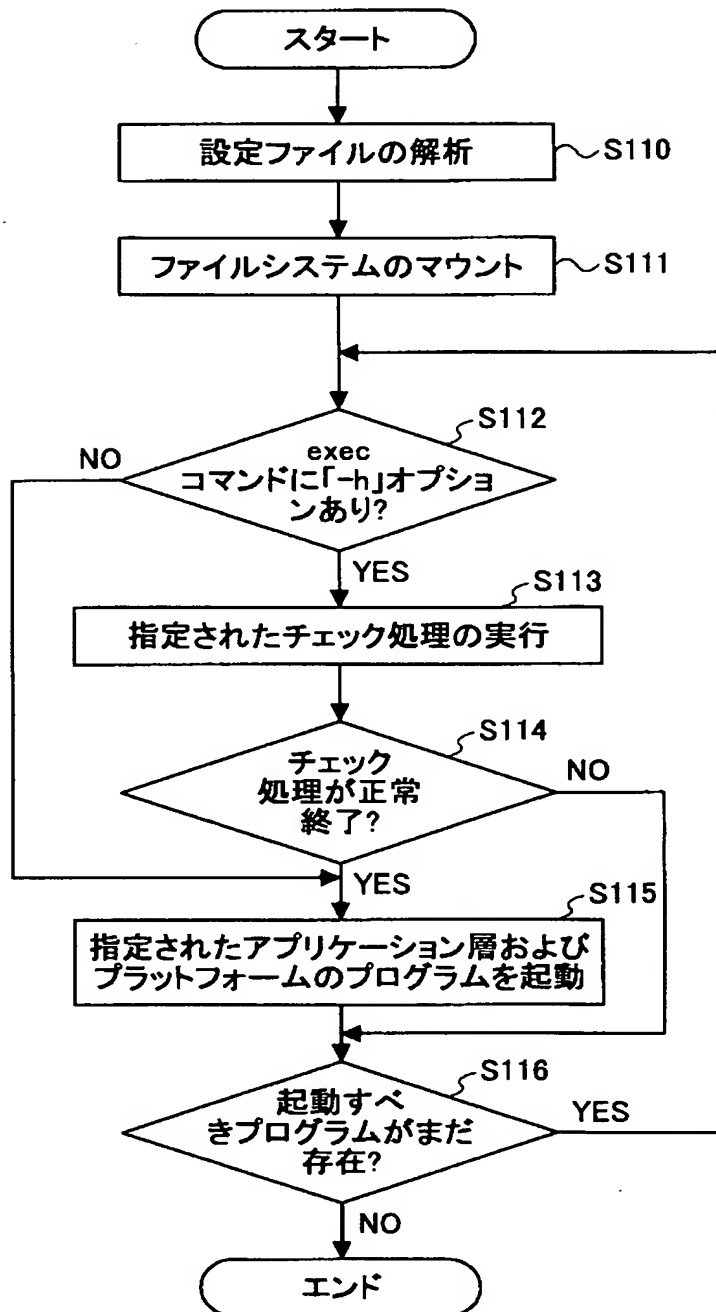
【図 4】

融合機起動部の処理の一例のフローチャート



【図 5】

プログラム起動部の処理の一例のフローチャート



【図 6】

設定ファイルの一例の構成図

```
exec -h fcuccheck /fax/bin/fax
```

```
exec -h cpuccheck1 /printer/bin/setfont_bitmap
```

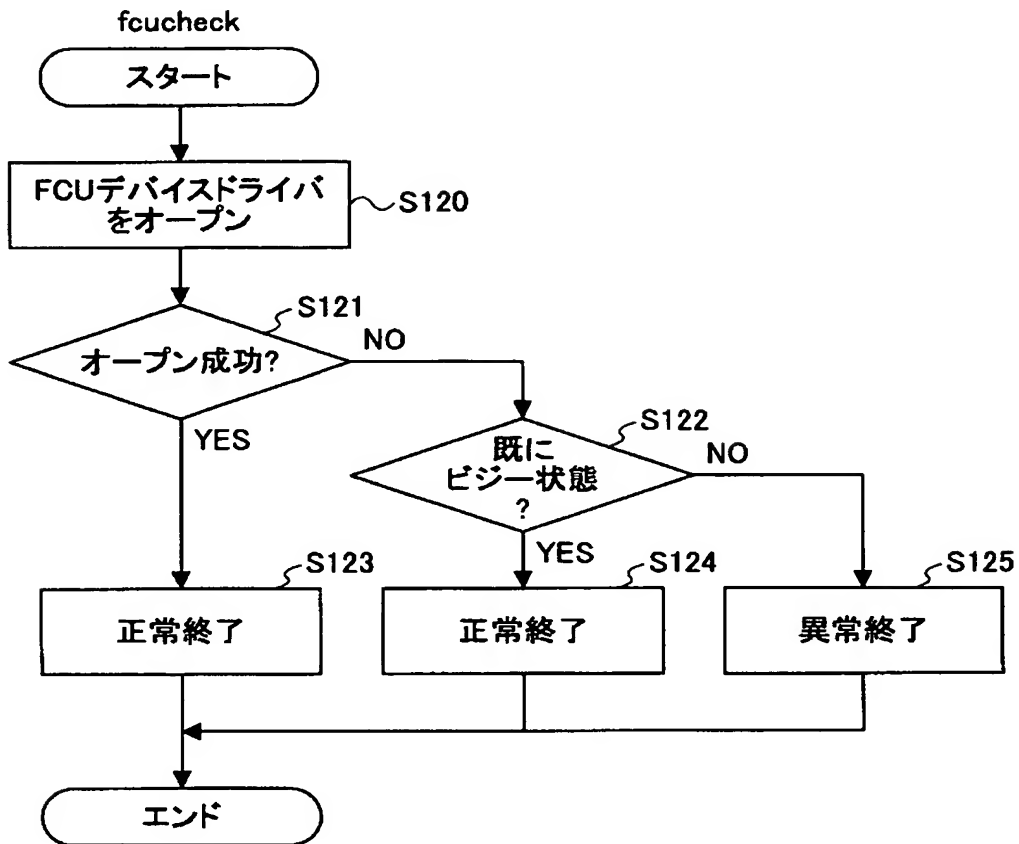
```
exec -h cpuccheck2 /printer/bin/setfont_vector
```

```
exec -h memcheck1 /web/bin/httpd -5
```

```
exec -h memcheck2 /web/bin/httpd -10
```

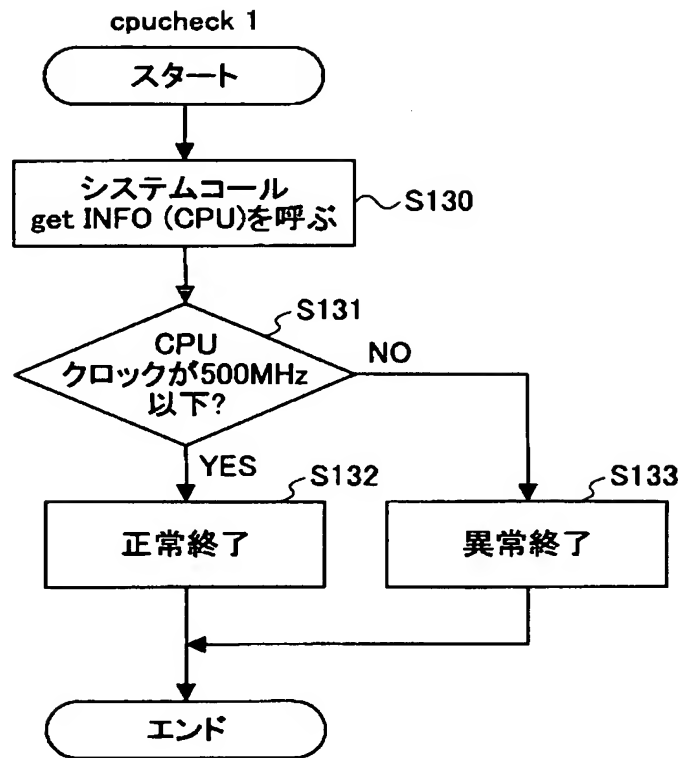
【図 7】

チェック処理「fcucheck」の一例のフローチャート



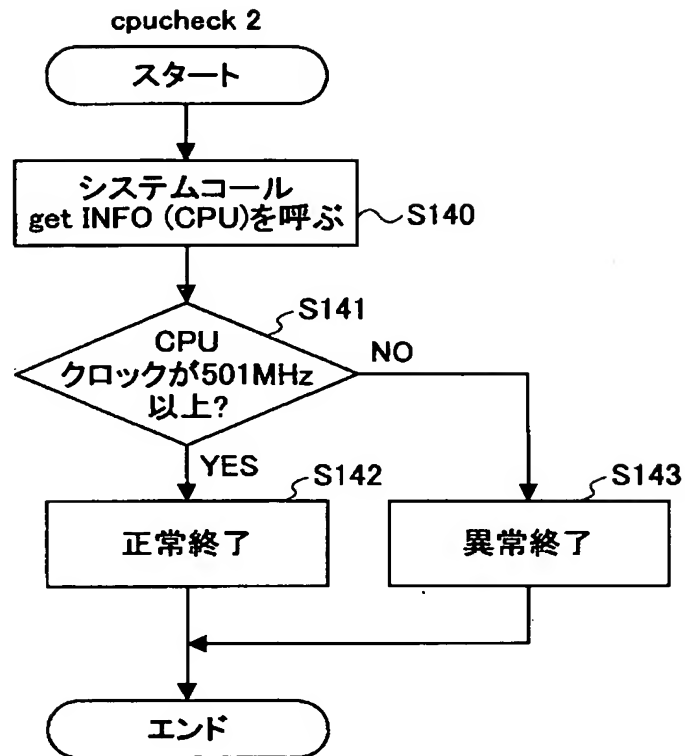
【図 8】

チェック処理「cpucheck 1」の一例のフローチャート



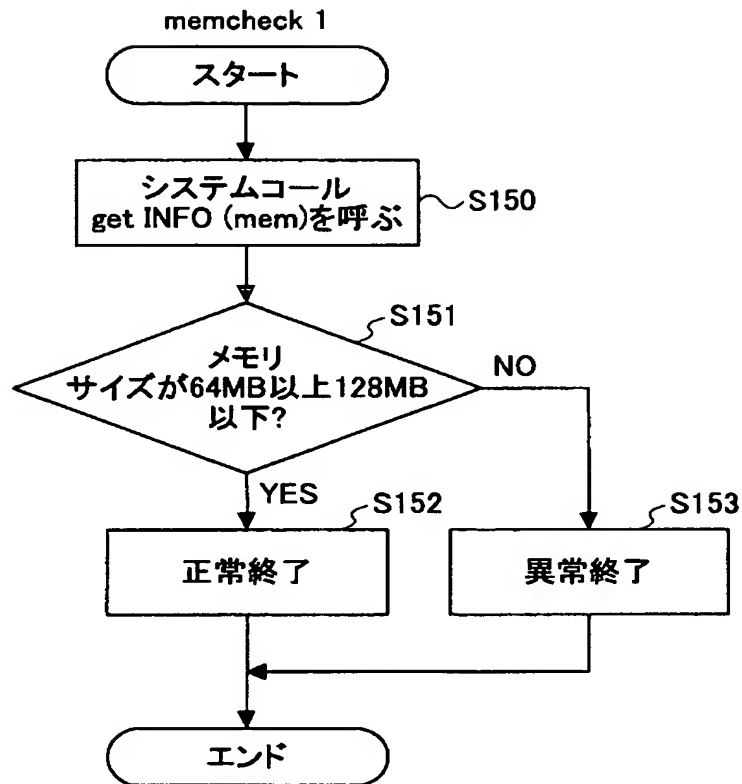
【図 9】

チェック処理「cpucheck 2」の一例のフローチャート



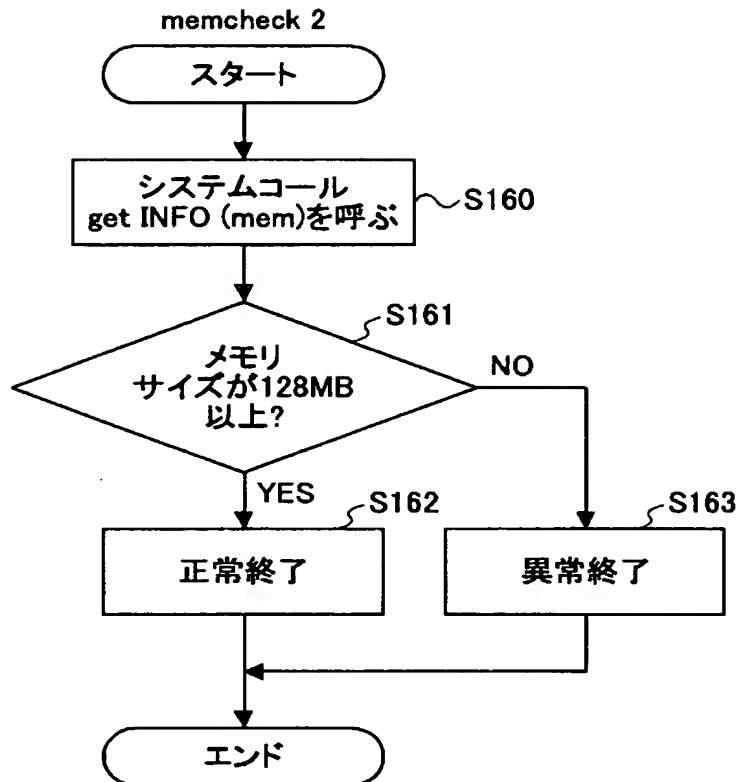
【図10】

チェック処理「memcheck 1」の一例のフローチャート



【図 11】

チェック処理「memcheck 2」の一例のフローチャート



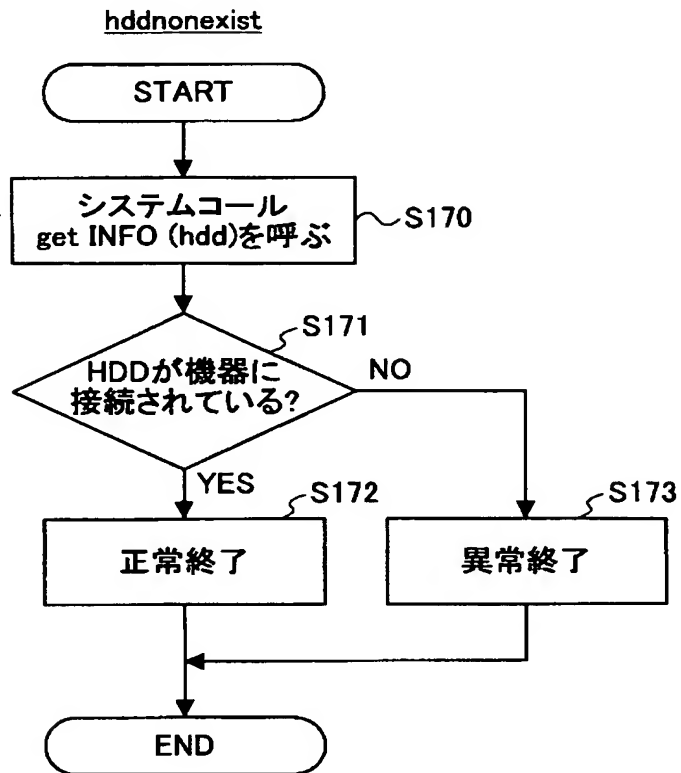
【図 12】

設定ファイルの他の一例の構成図

```
mount -h hddnonexist ramdisk /dev/md0c /ramdisk
```

【図 13】

チェック処理「hddnonexist」の一例のフローチャート



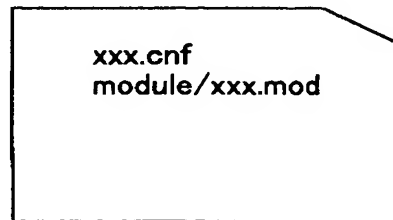
【図 14】

設定ファイルの他の一例の構成図

```
sd2
mount -h /sbin/sdcommand gzromfs xxx.mod /mnt
exec /mnt/xxx
```

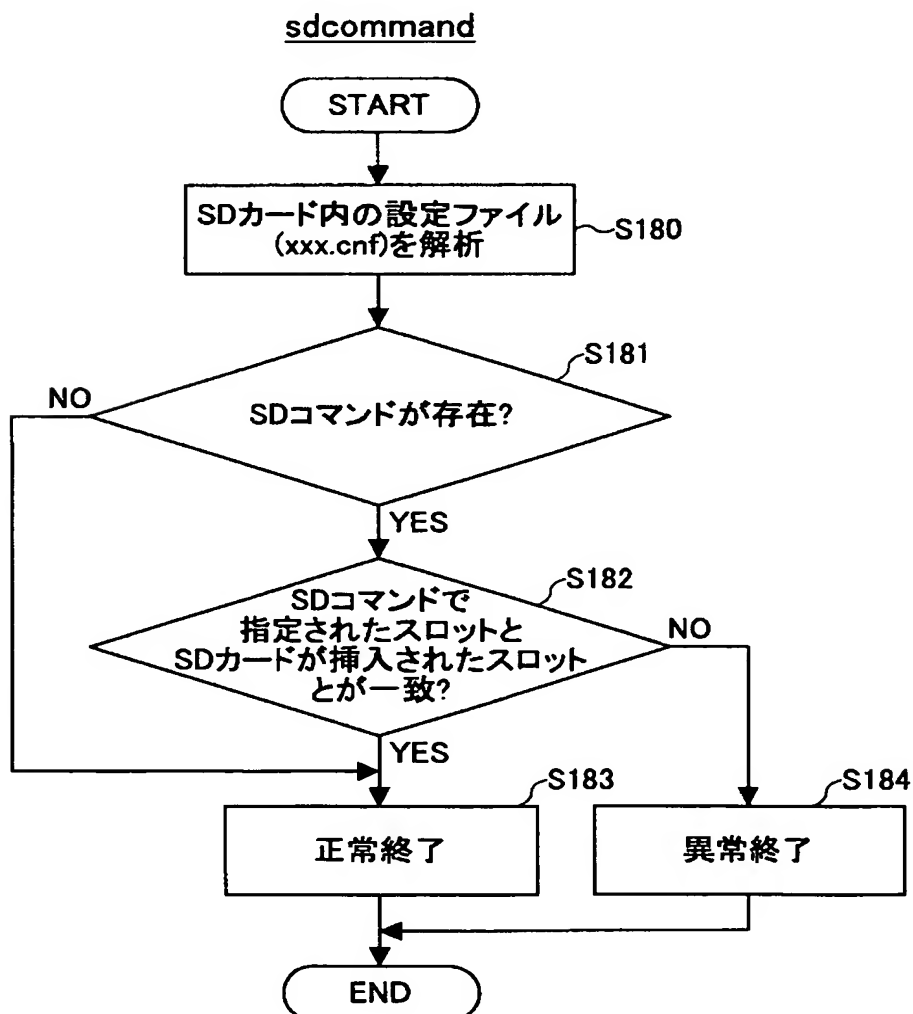
【図 15】

SDカード内に格納されているファイルの一例のイメージ図



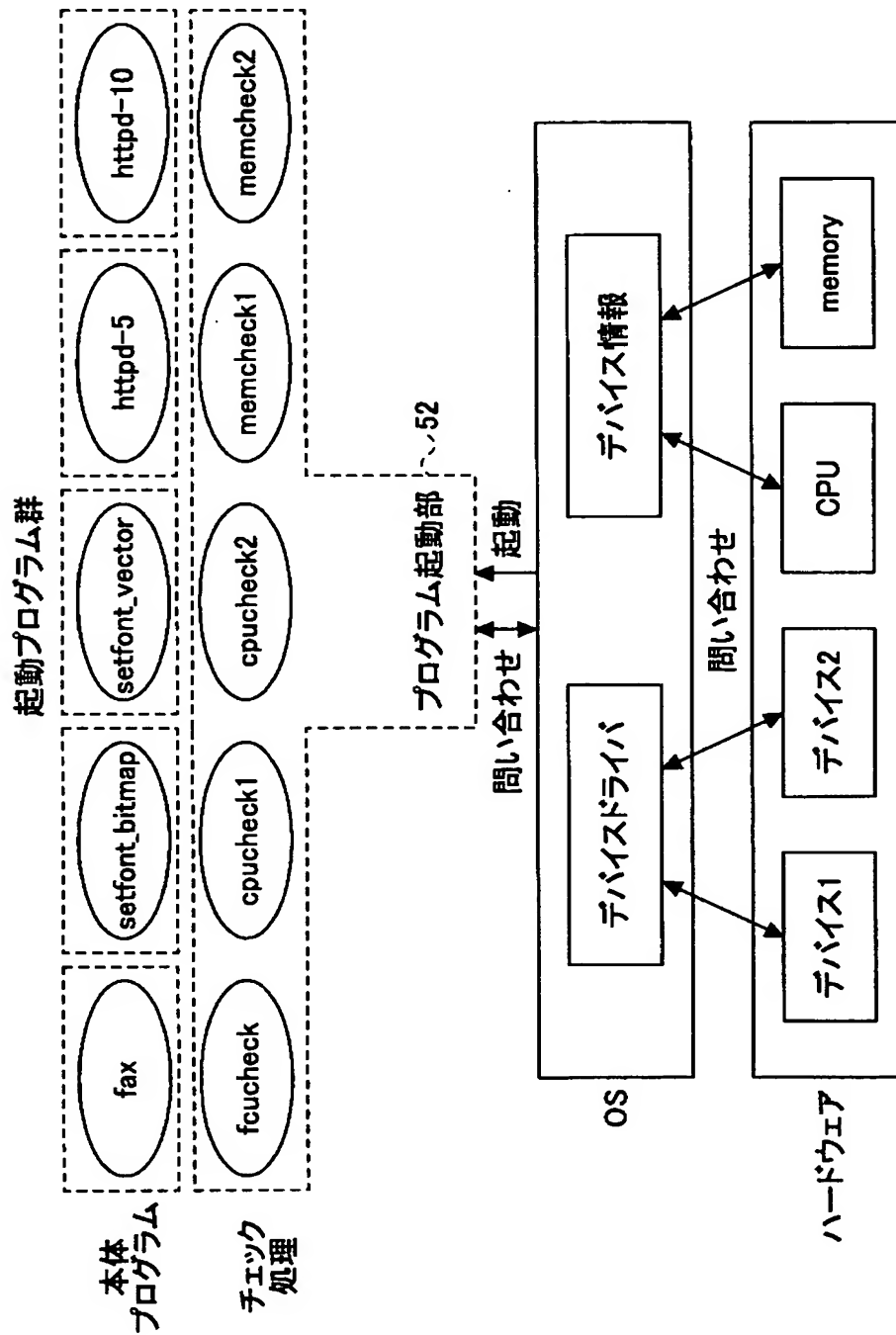
【図 16】

チェック処理「sdcommand」の一例のフローチャート



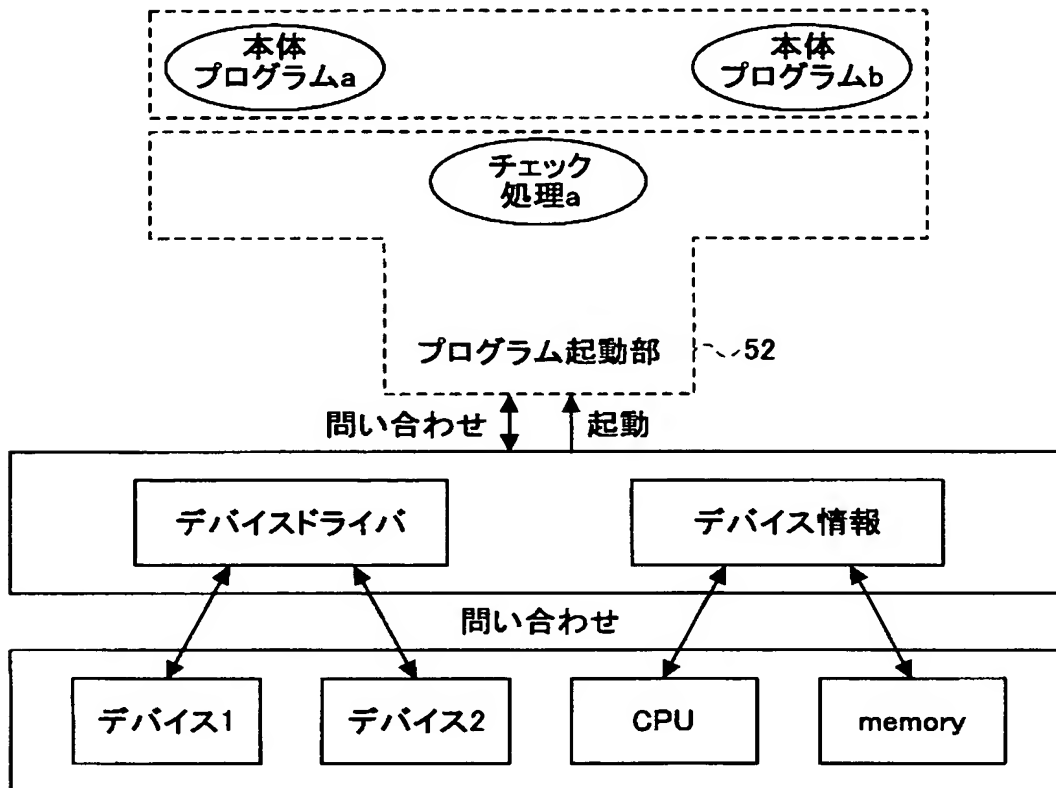
【図 17】

本体プログラムと、チェック処理と、プログラム起動部と、OSと、ハードウェアとの関係を表した一例の関係図



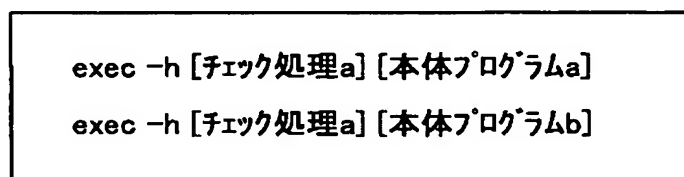
【図 18】

チェック処理と本体プログラムとが
1対nに対応する例について説明するための図



【図 19】

設定ファイルの他の一例の構成図



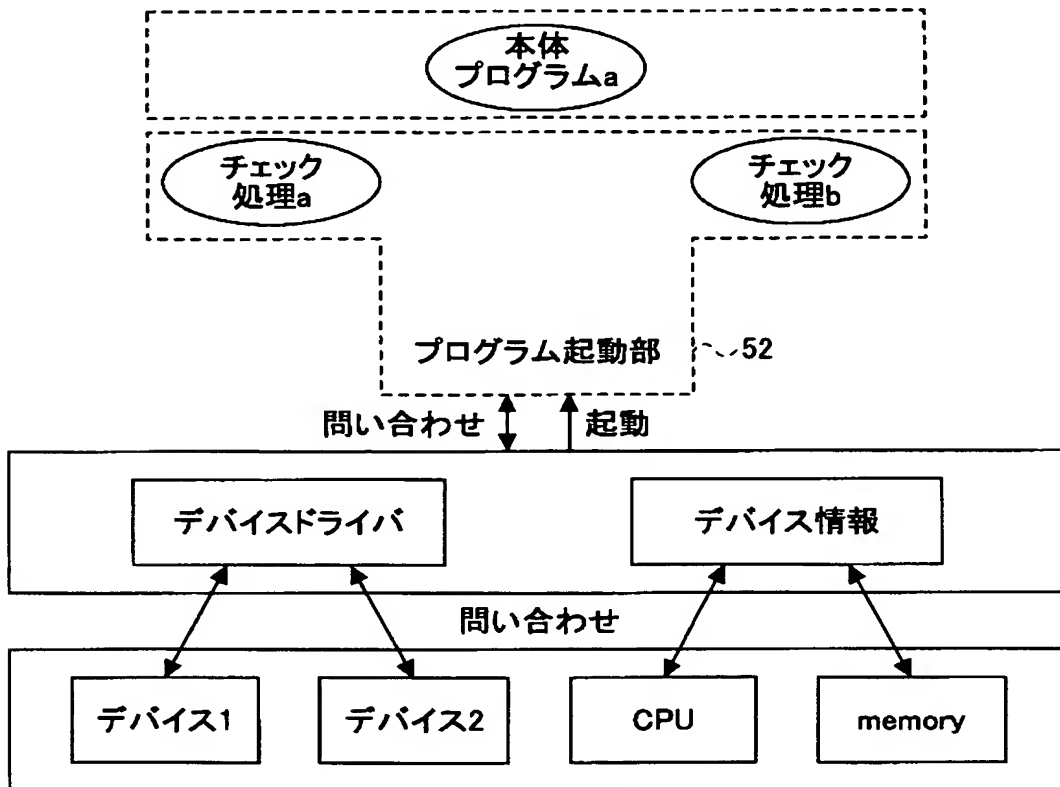
【図 20】

マウントを抑制する設定ファイルの一例の構成図

```
mount -h memcheck3 romfs web.romfs /web
```

【図 21】

チェック処理と本体プログラムとが
n 対 1 に対応する例について説明するための図

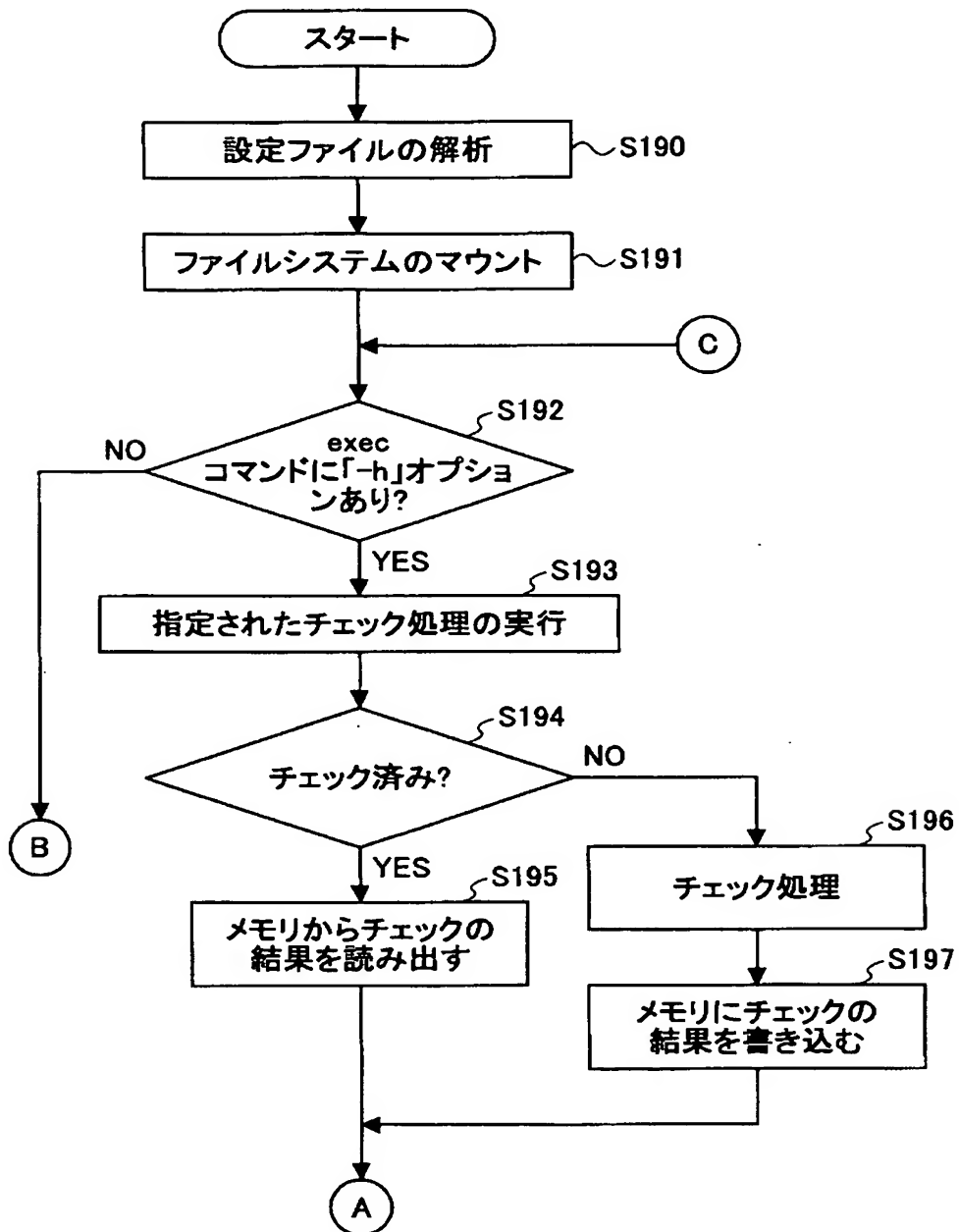


【図 2 2】

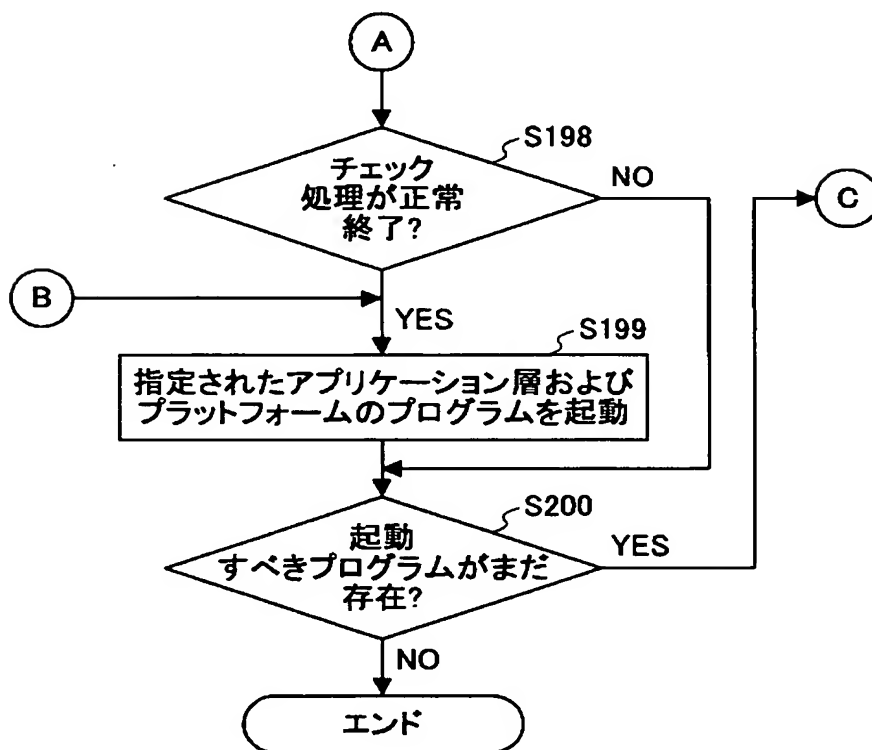
設定ファイルの他の一例の構成図

exec -h [チェック処理a]-h [チェック処理b] [本体プログラムa]

【図 23】

プログラム起動部が行うチェック処理の
一例のフローチャート (1/2)

【図 24】

プログラム起動部が行うチェック処理の
一例のフローチャート (2/2)

【書類名】 要約書

【要約】

【課題】 各プログラムの重複部分を削減することができ、ハードウェア資源に関連するプログラムを効率良く起動することが可能な画像形成装置およびプログラム起動方法を提供することを目的とする。

【解決手段】 画像形成処理で使用されるハードウェア資源と、画像形成に係る処理を行うプログラムとを有する画像形成装置であって、ハードウェア資源に関するチェックを行うチェック処理とプログラムとの関連を設定する設定手段と、チェック処理を行い、そのチェック処理の結果に応じてチェック処理に関連するプログラムを起動する起動手段 52 とを有することにより上記課題を解決する。

【選択図】 図 17

特願 2 0 0 3 - 3 9 3 4 1 6

出 願 人 履 歴 情 報

識別番号

[0 0 0 0 0 6 7 4 7]

1. 変更年月日

2 0 0 2 年 5 月 1 7 日

[変更理由]

住所変更

住 所

東京都大田区中馬込 1 丁目 3 番 6 号

氏 名

株式会社リコー